Kokeita symbolijonojen käytöstä käsin kirjoitettujen merkkien tunnistamisessa

Matti Aksela

22. helmikuuta 1999

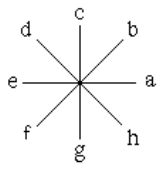
Tiivistelmä

Tässä työssä on tutkittu symbolijonojen käyttöä käsin kirjoitettujen merkkien tunnistamisessa eräitä vaihtoehtoisia menetelmiä kokeilemalla. Kokeissa on käytetty useita eri merkkijonojen muodostamistapoja sekä useita kustannusfunktiomalleja eri painoarvoilla symbolijonojen Levenshtein-etäisyyden laskemisessa. Luokittelu on tapahtunut symbolijonoille laskettujen etäisyyksien perusteella. Tulokset osoittavat, että symbolijonojen käyttö luokittelussa säästää kyllä huomattavasti laskentakapasiteettia mutta tunnistustulokset eivät yleensä ole riittävän hyviä käyttöä ajatellen.

1 Johdanto

Symbolijonojen käyttö merkintunnistuksessa on varsin yleinen lähestymistapa. Symbolijonoilla on joitain selkeitä etuja esimerkiksi pisteittäiseen sovitukseen verrattuna. Ennen kaikkea pienimuotoisemmissa laitteistoissa tulee merkittäväksi huomattava laskentakapasiteetin tarpeen väheneminen, koska symbolijonoja käyttämällä saadaan tiedon määrä huomattavasti pienemmäksi [6].

Perusajatuksena symbolijonon muodostamisessa on tehdä jonosta koordinaatteja jono symboleja, jotka kuvaavat mielellään yksikäsitteisesti koordinaattijonon eli tässä tapauksessa käsin kirjoitetun merkin. Symbolijono muodostetaan diskretoimalla joillakin arvoilla sekä suuntaa että etäisyyttä. Usein on valittu jokin vakiomitta symbolin esittäessä suuntaa, mutta on myös mahdollista sisällyttää pituustieto koodiin. Tällöin joko koodataan symbolit niin, että niihin mahtuu diskretoidun suunnan lisäksi tieto diskretoidusta pituudesta, tai käytetään rinnakkain kahta jonoa, joista toinen sisältää diskretoidut suunnat ja toinen vastaavat pituudet. Tässä työssä on käytetty sekä



Kuva 1: Symbolit 8-suuntaisessa diskretoinnissa.

vakiomittaisia symboleja että symboleja, joihin on liitetty diskretoimaton pituus.

2 Menetelmät

2.1 Symbolijonon luominen

Perusajatuksena on muodostaa vaihtelevilla etäisyyksillä toisistaan sijaitsevista koordinaattipareista symbolijono, johon voi lisäksi kuulua etäisyyskoodi. Kokeita on suoritettu sekä käyttäen etäisyyskoodia että pelkällä suuntatiedolla. Erityisesti pelkkää suuntatietoa hyödynnettäessä on oleellista, että käytössä on joku tietty diskretointietäisyys, jonka pituisia symboleja vastaavat vektorit sitten ovat. Tässä työssä on kokeiltu useita eri etäisyyksiä. Ennen symbolijonon luomista kaikki merkit skaalattiin ympäröivän nelikulmion pisimmän sivun mukaan samankokoisiksi.

Diskretointisuuntien määrää on myös vaihdeltu, käytännössä välillä 4–32 kahden potensseina. Diskretoitaessa on siis otettu ensin lähtöpiste ja liikuttu kynän jälkeä eteenpäin kunnes on saavutettu tarvittava etäisyys alkupisteestä tai tultu jäljen loppuun. Suuntien diskretoidut lukuarvot muutettiin merkkijonon selkeyden vuoksi kirjainsymboleiksi $\{abcde\ldots\}$. Kuvassa 1 on esitetty kuvio 8-suuntaisen diskretoinnin symboleista.

Osassa ajoja käytössä on ollut kulmien tunnistusta varten erillinen osa, jonka avulla pyritään ottamaan jyrkät muutokset käyrässä huomioon aproksimoimalla kulmaa ja sen derivaattaa kaavan (1) mukaisesti. Tämä yhtälö aiheuttaa jonkin verran painotusta napakoordinaatistossa lähempänä origoa tapahtuville muutoksille. Kokeiltaessa tämän vaikutusta tunnistustuloksiin havaittiin sen antavan myös jonkin verran parempia tuloksia kuin pelkään derivaattaan perustuvan kulmien tunnistuksen, sillä käsinkirjoitetuissa merkeissä tunnistuksen kannalta merkittävimmät muutokset tapahtuvat yleensä

Alkuperäinen merkki	cdlldo	cdlldhna
		
Alkuperäinen merkki	fffbeeghh	fffe"eeghh
		

Kuva 2: Esimerkkejä symbolijonoista.

juuri origon lähellä.

$$\phi_i = \arctan(\frac{x_{i-2} + x_{i-1} - x_i}{y_{i-2} + y_{i-1} - y_i}) \tag{1}$$

ϕ_i:n arvot laskettiin pisteissä i ja $i-3$, ja näiden erotuksen $\Delta\phi$:n kasvaessa yli raja-arvon, tulee i:sta seuraava diskretointipiste. Raja-arvona käytettiin arvoa π/d, missä d on diskretointisuuntien määrä.

Toinen kokeiltu variaatio on ollut erottaa vektorit, jotka piirtyvät kynän ollessa irti piirtoalustasta. Tällöin vektoreita luonnollisesti muodostuu vain, kun vedot on yhdistetty, mutta vetojen yhdistäminen tuntuu järkevimmältä ratkaisulta symbolijonoja käsiteltäessä. Lisäksi se vähentää jonkin verran tarvittavien prototyyppien määrää, koska eri määrän vetoja sisältävät merkit oli tässä tutkimuksessa alunperin määritelty olemaan äärettömän etäisyyden päässä toisistaan. Tässä on päädytty merkitsemään kynän noston ja laskun välistä matkaa joukon $U = \{!, \#, \$, \%, \&, \dots\}$ symboleilla.

Kuvassa 2 on esitetty kaksi käsin kirjoitettua merkkiä ja niille molemmille kaksi erilaista kuvausta. Merkin “f” kohdalla ensimmäisessä muodossa ei ole käytetty kulmien eikä kynän noston huomiointia, ja toisessa taas jyrkät suunnanmuutokset on otettu huomioon. Tämän merkin symbolijonoesityksen muodostamisessa on käytetty 16-suuntaista diskretointia, ja yhden symbolin oletuspituus on 30. Merkin “k” kohdalla ensimmäinen muoto kuvaa ilman kynän noston huomiointia muodostettua merkkiä ja toisessa nosto on otettu huomioon. Tähän merkkiin taas on sovellettu 8-suuntaista diskretointia symbolinpituudella 15. Koska nostoa huomioitaessa kynän nosto- ja laskupisteiden väli on yksi symboli, on symbolijonoissa havaittavissa pientä eroa muuallakin kuin tämän nimenomaisen symbolin kohdalla.

	0	1	2	3	...	n
0		W	o	r	d	1
1	W	$d(a^1, b^1)$	...	...	...	...
2	o	⋮	⋱			
3	r	⋮		⋱		
⋮	d	⋮			⋱	
m	2	⋮				Result

Taulukko 1: Etäisyyksien laskentataulukko.

2.2 Etäisyyden laskeminen

2.2.1 Levenshtein-etäisyys

Symbolijonojen etäisyyksien laskemisen perustana on käytetty Levenshtein-etäisyyttä, joka on määritelty seuraavasti [4]. On olemassa kolme sallittua operaatiota:

1. korvaus $x \to y$
2. x:n poisto
3. y:n lisays

Näille kaikille määritellään omat kustannusfunktionsa, jotka voidaan merkitä yhtenevästi, kun ϕ-merkillä tarkoitetaan tyhjää:

1. korvaus: $w(x, y)$
2. poisto: $w(x, \phi)$
3. lisäys: $w(\phi, y)$

Kokeiluissa on käytetty sekä etäisyysfunktioiden perusmallia että muotoa, jossa lisäys ja poisto riippuvat lisättävän/poistettavan merkin lisäksi myös muista sen vieressä olevista merkeistä.

Minimietäisyyden laskemiseksi muodostetaan yllämainittujen kustannusfunktioiden avulla taulukon 1 kaltainen taulukko, jossa jokaisen ruudun arvo $d(a^i, b^j)$ on:

$$d(a_i, b_j) = \min \begin{cases} d(a_{i-1}, b_j) + w(a_i, \phi) & \text{(lisäys)} \\ d(a_{i-1}, b_{j-1}) + w(a_i, b_j) & \text{(korvaus)} \\ d(a_i, b_{j-1}) + w(\phi, b_j) & \text{(poisto)} \end{cases} \tag{2}$$

Etäisyys $d(a_0, b_0) = 0$ ja etäisyyksien laskenta tehdään laskien ensin ensimmäinen rivi ja sarake, koska niissä on vain yksi vaihtoehto, ylimmällä rivillä poisto ja ensimmäisessä sarakkeessa lisäys. Muut lasketaan yhtälön (2) mukaisesti.

2.2.2 Käytetty etäisyysmäärittely

Tässä työssä etäisyys on määritelty yllä annettuun Levenhstein-etäisyyteen pohjautuen. Kustannusfunktioiden rakennetta on kuitenkin jonkin verran muokattu paremmin juuri tähän sovellukseen sopivaksi. Käytetyt kustannusfunktiot w voidaan jakaa seuraaviin luokkiin:

1. diskretoitujen suuntien erotuksesta lineaarisesti riippuva etäisyys
2. sekä suuntien että pituuksien erotuksesta lineaarisesti riippuva etäisyys
3. sekä suuntien että pituuksien erotuksesta painotetusti lineaarisesti riippuva etäisyys
4. Lisäykselle ja poistolle voidaan asettaa lisäksi määrittely: $w(y, \phi) = w(\phi, y) = 0$, jos symboli y on sama kuin sen vasemmalla tai oikealla puolella oleva symboli, muuten $w(y, \phi) = w(\phi, y) = 1$

Sinänsä nämä kaikki ovat Levenshtein-etäisyysmäärittelyyn sopivia, mutta apuna on lisäksi käytetty tietoa ympäröivistä merkeistä. Lisäys on sitä kalliimpi mitä enemmän lisättävä tai poistettava merkki poikkeaa lähiympäristöstään. Lopullisissa kokeissa on todettu kohdan neljä mukaisella määrittelyllä saatavan parhaita tuloksia, ja sitä on käytetty. Lopullinen etäisyysmalli oli muotoa:

$$d(a^i, b^j) = \min \begin{cases} d(a^{i-1}, b^j) + w_l(a_{i-1}, b_j, a_{i+1}) & \text{(lisäys)} \\ d(a^{i-1}, b^{j-1}) + w_k(a_i, b_j) & \text{(korvaus)} \\ d(a^i, b^{j-1}) + w_p(a_{i-1}, a_i, a_{i+1}) & \text{(poisto)} \end{cases} \tag{3}$$

Merkitään diskretointisuuntien lukumäärää d:llä, diskretointipituutta ℓ:llä ja yksittäisen merkin pituutta ℓ_{a_j}:lla. Lisäksi merkitään lisäyksen vakioita λ_i:llä, korvauksen vakioita κ_i:llä sekä poiston vakiota ρ:lla. Joukolla U merkitään kynä-ylhäällä-symboleita.

Kaavan (4) $w_l(a_{i-1}, b_j, a_{i+1})$ on lisäyksen määrittelevä funktio. Laskennan pohjana on kaikkien lisäysten kustannuksiin liitettävä lisäyksen hinta λ_1, joka kerrotaan pituuksia käytettäessä pituuden ja diskretointietäisyyden suhteella (jos pituuksia ei käytetä, suhde on yksi). Lisäksi otetaan huomioon

onko lisättävä merkki sama jomman kumman tai molempien viereistensä merkkien kanssa. Jos näin on, lisäyksen kustannus alenee vastaavasti λ_2:n tai λ_3:n verran. Jos kynän nostot huomioidaan, ja lisättävä merkki olisi kynä-ylhäällä-symboli, kustannukseen lisätään kynä-ylhäällä-symbolin lisäyksen hinta λ_4.

$$w_l(a_{i-1}, b_j, a_{i+1}) = \lambda_1 \frac{\ell_{b_j}}{\ell} + \begin{cases} -\lambda_2, \text{ jos } b_j \subset \{a_{i-1}, a_{i+1}\}, a_{i-1} \neq a_{i+1} \\ -\lambda_3, \text{ jos } b_j = a_{i-1} = a_{i+1} \\ \lambda_4, \text{ jos } b_j \subset U \\ 0, \text{ muulloin} \end{cases} \tag{4}$$

Yhtälön (5) $w_k(a_i, b_j)$ on korvauksen hinnan laskemiseen käytettävä funktio, jonka pohjana on a_i:n ja b_j:n pituuksien erotuksen suhde ℓ:ään kerrottuna κ_1:llä. Jälleen jos pituuksia ei ole käytössä, pidetään suhdetta yhtenä. Tähän lisätään merkkien suuntien välinen erotus ja jos kynän noston huomionti on käytössä, on κ_2 kynä-ylhäällä-symbolin korvaamiseen määritelty hinta.

$$w_l(a_i, b_j) = \kappa_1 \frac{|\ell_{a_i} - \ell_{b_j}|}{\ell} + \frac{\angle(a_j - b_j)}{2\pi} + \begin{cases} \kappa_2, \text{ jos } a_i \subset U, b_j \not\subset U, \\ \quad \text{tai } a_i \not\subset U, b_j \subset U \\ 0, \text{ muulloin} \end{cases} \tag{5}$$

Poiston kustannuksen määrittelevä funktio $w_p(a_{i-1}, a_i, a_{i+1})$ on käytännössä sama kuin lisäyksen määrittelevä funktio, mutta sille on annettu painokerroin ρ jolla saadaan säädettyä poistojen ja lisäysten hintojen välistä suhdetta.

$$w_p(a_{i-1}, a_i, a_{i+1}) = \rho \, w_l(a_{i-1}, a_i, a_{i+1}) \tag{6}$$

Lopullisen etäisyyden laskeminen tapahtuu taulukon 1 tapaisen taulukon avulla. Jokaiseen ruutuun lasketaan kaavan (2) avulla kertymäkustannus, joka riippuu aiemmista kustannuksista. Näin ollen viimeiseen ruutuun saadaan koko symbolijonojen erotus. Tässä laskenta on suoritettu kokonaisuudessaan tällä varsin yksinkertaisella tavalla. Tehdyissä kokeissa on pyritty selvittämään parhaat kustannusfunktiot sekä diskretointisuuntien määrän, diskretointietäisyyden sekä muiden parametrien varioinnin vaikutusta.

Etäisyyden laskennassa varioitavina parametreina olivat siis seuraavat:

1. diskretointisuuntien lukumäärä d
2. diskretointivälin pituus ℓ
3. korvauksen hinta pituuskoodatussa symbolijonossa κ_1
4. lisäyksen hinta pituuskoodatussa symbolijonossa λ_1

5. poiston ja lisäyksen hintojen suhde ρ
6. kynä-ylhäällä-symbolin lisäyksen hinta λ_2
7. kynä-ylhäällä-symboliksi muuttamisen hinta κ_2

Näistä voidaan olettaa, että korvauksen hinnan tulee olla aika lähellä lisäyksen ja poiston summaa, todennäköisesti hieman alle, jottei kumpaakaan vaihtoehtoa suosittaisi liikaa. Korvaushan vastaa lisäystä ja poistoa suoritettuina peräkkäin. Voidaan olettaa, että poiston ja lisäyksen kustannusten tulisi olla hyvinkin samanlaisia, mutta käytännössä tämä riippuu luokiteltavista merkeistä ja prototyypeistä, eli vaihtelua voi olla jonkin verran.

Kynän noston kustannuksen ollessa hyvin pieni vetojen lukumäärän vaikutus luonnollisesti vähenee, kynä ilmassa piirretyn vektorin lisäyksellähän on sama merkitys kuin vetojen lukumäärän kasvattamisella. Tehdyissä kokeissa havaittiin, että vetojen yhdistäminen heikentää tällä järjestelmällä tunnistustulosta hieman. Näin ollen voidaan olettaa, että parhaita tuloksia saataisiin nostamalla vetomäärän kasvattamiseen käytettävien kynä-ylhäällä-symbolien kustannusta hieman. Tämä vähentäisi vetoja yhdistävää kynä-ylhäällä-symbolien korvausta tavallisilla suuntasymboleilla. Lisäksi tarpeeton kynä-ylhäällä-symbolien lisääminen ja merkkien pilkkoutuminen vähenisi.

2.2.3 Muita etäisyysmäärittelyjä

Voidaan tietenkin ajatella, että korvaus, poisto ja lisäys ovat vain osa mahdollisista merkkijono-muunnoksista, kaikkiaan toimenpiteitä voidaan ajatella olevan kuudenlaisia [4]:

1. korvaus
2. lisäys
3. poisto
4. tiivistäminen
5. laajentaminen
6. siirto

Näistä tiivistäminen ja laajentaminen voidaan kuitenkin helposti ajatella olevan vain lisäyksen ja poiston erikoistapauksia, eli lisätään tai poistetaan

sama merkki kuin viereinen. Tehdyissä kokeissa tätä onkin tutkittu ja otettu huomioon merkkien samankaltaisuus. Tiivistämisellä ja laajentamisella olisikin varsin suuri merkitys, jos ei suoritettaisi kokoskaalausta ennen merkin tunnistamista. Tällöinhän halpojen tiivistys- ja laajennusoperaatioiden avulla voitaisiin olettaa skaalan merkityksen melkolailla häviävän. Koska nyt tehdyssä koesarjassa käytettiin valmiiksi skaalattuja merkkejä, näiden operaatioiden merkitys on pienentynyt.

Siirrolla tarkoitetaan kahden, yleensä peräkkäisen, merkin paikan vaihtamista. Tämä tapaus onkin yleisesti käytännöllinen esimerkiksi konekirjoitetun tekstin korjaamisessa, jossa kahden merkin paikan vaihtuminen on varsin yleinen näppäilyvirhe [4]. Tämän työn tapauksessa symbolien järjestyksen vaihtumista ei voida pitää mitenkään niiden vaihtamista lievempänä tapahtumana, joten tätä ei erikseen ole käsitelty.

Yllä estetyn Levenshtein-mitan lisäksi yleisesti käytetään ainakin kolmea etäisyysmittaa [4]. Niistä merkittävimmät ovat:

1. *dynamic time warping*-menetelmän yhteydessä käytettävät mitat, joissa juuri tiivistysten ja laajennusten merkitys kasvaa huomattavasti,
2. suora yleistys Levenshtein-mitasta, joka näissäkin kokeissa on useimmiten ollut käytössä, Levenshtein-etäisyyden määrittelyjä on muokattu jonkin verran sopimaan paremmin kyseiseen sovellukseen,
3. suuntia esittävien symbolijonojen muodostamien kokonaisuuksien ja niiden yhdistelmien tutkiminen.

3 Toteutus

Tässä erikoistyössä esitettävät tulokset on saatu liittämällä symbolijonojen käsittely yhdeksi mahdolliseksi etäisyydenlaskutavaksi käsin kirjoitettujen merkkien tunnistamista varten kehitettyyn järjestelmään [2, 3, 5]. Käytännön toteutus on tehty C++-kielisenä ohjelmana UNIX-ympäristössä. Merkkijonojen luonti on tapahtunut sen jälkeen, kun merkin koko on normalisoitu esikäsittelyllä. Näin diskretointia tehtäessä tiedetään etukäteen merkin ulottuvuudet.

Optimaalisten parametrien etsimiseksi suoritettiin ajoja varsin pienellä datamäärällä, minkä jälkeen lupaavia tuloksia antaneita parametriyhdistelmiä kokeiltiin myös suuremmalla määrällä luokiteltavia merkkejä. Yleensä kaikissa ajoissa merkkien prototyypit valittiin klusteroimalla käyttäen samoja parametriarvoja kuin luokittelussakin. Siten prototyyppijoukko oli jokaisessa tapauksessa käytettävälle luokittelijalle optimaalinen.

Merkit	Opetusjoukko		Testijoukko	
	Kirjoittajia	Lukumäärä	Kirjoittajia	Lukumäärä
a-z	114	2923	10	260
a-öA-Ö0-9	114	7231	10	665

Taulukko 2: Ensimmäisen datajoukon merkkien frekvenssit.

4 Kokeet

Kokeita on tehty kahdella huomattavan erilaisella datajoukolla. Ensimmäisissä kokeissa kerätyt merkit olivat selvästi pienempikokoisia ja vähemmän informaatiota sisältäviä kuin toisissa kokeissa käytetyt. Lisäksi ensimmäisessä koesarjassa ei käytetty minkäänlaista klusterointimenetelmää, vaan prototyyppijoukkona käytettiin kaikkia opetusjoukon merkkejä.

4.1 Kokeet ensimmäisellä datajoukolla

4.1.1 Datajoukon kuvaus

Ensimmäisissä kokeissa on ollut käytössä dataa, jota on kerätty noin 40 pistettä/s-taajuudella. Paikkaresoluutio on ollut luokkaa 10 pistettä/mm. Kynän piirtonopeuden, aika- ja paikkaresoluution yhteisvaikutuksesta johtuen pisteiden päällekkäisyyttä esiintyy datajoukossa hyvin paljon. Lisäksi datasta puuttuu sekä paine- että aikainformaatio. Taulukossa 2 on esitetty merkkijoukkojen koko.

4.1.2 Kokeiden suoritus

Kokeet suoritettiin käyttämällä koko opetusjoukkoa testijoukon luokitteluun. Käytössä oli yleinen k-NN-menetelmä [1], mutta osoittautui, että parhaat tulokset saatiin kuitenkin parametriarvolla $k = 1$. Käytännössä merkin luokitus siis tapahtui etsimällä sille vertailujoukon keskuudesta samanlaisin merkki, jonka luokkaa sitten käytettiin tunnistustulokseen. Kokeissa käytettiin seuraavia esikäsittelytoimenpiteitä:

1. pisimmän sivun skaalaus sadan pituiseksi siten, että sivujen suhteet säilyvät,
2. rajaavan nelikulmion mukainen siirto origon ympärille,
3. päällekäisesti toistuvien pisteiden poisto.

Esikäsittelytoimenpiteet ovat sinänsä varsin selkeitä ja perusteltuja. Skaalaus on välttämätön, jotta kirjoittajien käsialan koko ei vaikuttaisi tuloksiin, ja sivunpituudeksi valittu 100 johtuu vain tämän luvun käsittelyn helppoudesta symbolijonon luonti- ja luokittelurutiinien toteuttamisessa. Siirto origon ympärille on myös välttämätön, koska ei voida olettaa kirjoittajien kirjoittavan aina samaan kohtaan. Useampikertaisten pisteiden poisto taas on merkkijonoja muodostettaessa tarpeen, koska symbolit ilmoittavat differenssejä pisteiden välillä.

Ensimmäisissä kokeissa kokeiltiin useita eri symbolijonon muodostusparametrien arvoja ja hintoja etäisyyden laskemisessa tavoitteena saavuttaa pienin mahdollinen virheprosentti. Kokeet suoritettiin käytännössä käymällä parametriavaruutta läpi varioimalla kaikkien parametrien arvoja järkevällä alueella, eli suoritettiin muutama tuhat ajoa. Hyvän tuloksen löytyessä tarkennettiin parametrien arvoja läheiseltä alueelta. Koko datan käyttö oli mahdollista, koska klusterointia ei tehty erikseen, ja testijoukon koko oli kuitenkin pieni.

4.1.3 Kokeiden tulokset

Tarkemmat tiedot luokittelijan parametreistä ja parhaat tulokset ovat taulukossa 3. LSS-luokitin on symbolijonoluokitin, joka käyttää suoraan etäisyysinformaatiota.

Kokeiden tuloksista voidaan nähdä, että luokittelutulos ei ole kovin loistava varsinkaan, kun käytössä ovat kaikki merkit. Tulos pienillä merkeillä on kohtalainen verrattuna muihin menetelmiin (Luku 5), mutta yli 13 prosentin virhe tekisi järjestelmän käytännössä mahdottomaksi käyttää.

4.2 Tulokset jälkimmäisellä datalla

4.2.1 Datan tiedot

Toinen käytetty datajoukko on Informaatiotekniikan laboratoriossa lokakuussa 1997 kerättyä dataa, jonka tarkkuus on selvästi parempi kuin ensimmäisen. Tallennetut merkit ovat selvästi suurempia kuin ensimmäisessä datajoukossa, ne koostuvat useammista pisteistä ja ensimmäisessä datajoukossa erittäin yleinen samojen koordinaattien toistuminen on huomattavasti harvinaisempaa. Tämän datajoukon keräyksessä aikaresoluutio on ollut noin 205 pistettä/s ja paikkaresoluutio noin 100 pistettä/mm. Lisäksi datajoukossa on selvästi eroteltu eri kirjoittajat ja jokaiselta on kerätty samat merkit. Mukana on myös alustalta saatu aika- ja paineinformaatio. Taulukossa 4 on

Luokittelija	**LSS**				
Merkkijoukko	a-z	a-öA-Ö0-9			
Prototyyppien kokonaismäärä	2923	7231			
Kynän noston huomionti	on	on	ei	on	on
Kulmien havainnointi	on	on	on	ei	on
Pituusinformaatio	on	on	on	on	ei
Suuntien lukumäärä d	32	32			
Diskretointivälin pituus ℓ	15	15			
Korvauksen lisähinta κ_1	0.55	0.6			
Lisäyksen lisähinta λ_1	0.05	0.2			
Poiston ja lisäyksen hintojen suhde ρ	1.1	1			
Kynä-ylhäällä korvauksen hinta κ_2	5	5			
Kynä-ylhäällä lisäyksen hinta λ_2	5	5			
Paras tulos	**13.2**	**35.3**	**36.1**	**44.5**	**44.8**

Taulukko 3: Ensimmäisessä kokeessa käytettyjen omien luokittimien parhaat tulokset.

Merkit	Kirjoittajia	Lukumäärä
a-z	22	8112
a-öA-Ö0-9	22	10626

Taulukko 4: Jälkimmäisen datajoukon merkkien frekvenssit.

esitetty merkkijoukkojen koko.

4.2.2 Kokeiden suoritus

Kokeet on suoritettu muodostamalla datasta ensin klusteroimalla samaa etäisyysmittaa käyttäen prototyyppijoukko, jossa oli taulukon 5 mukainen prototyyppijakauma. Samaa datajoukkoa, josta prototyypit oli tehty, käytettiin myös testaukseen. Tämä aiheuttaa jonkinasteista virhettä, koska samoja kirjoittajia ei periaatteessa saisi olla sekä opetus- että testijoukossa. Koska merkkejä kuitenkin oli erittäin paljon enemmän kuin prototyyppejä, voidaan olettaa, etteivät luokittelutulokset juurikaan poikkea todellisista. Kaikki tulokset vertailuosiossakin on saatu samalla menetelmällä, joten tulokset ovat sikäli keskenään vertailukelposia.

Kokeet suoritettiin käymällä parametriavaruutta läpi varioimalla kaikkien parametrien arvoja järkevällä alueella, eli pienellä datamäärällä suoritettiin tuhansia ajoja. Koko datajoukolla suoritettiin lopuksi ajoja vain muutamilla parametrien arvoilla löydetyn minimikohdan ympäriltä.

Esikäsittelymenetelmät tässä kokeessa olivat samat kuin edellisessäkin koe-

Pienet kirjaimet									
a	b	c	d	e	f	g	h	i	j
8	8	4	8	5	8	6	7	7	7
k	l	m	n	o	p	q	r	s	t
8	5	7	8	4	4	8	6	6	8
u	v	w	x	y	z	å	ä	ö	
6	6	7	7	8	6	10	7	8	
Isot kirjaimet									
A	B	C	D	E	F	G	H	I	J
7	6	4	6	6	6	7	8	4	3
K	L	M	N	O	P	Q	R	S	T
7	6	7	7	4	7	5	7	4	4
U	V	W	X	Y	Z	Å	Ä	Ö	
5	5	7	7	8	6	9	8	8	
Numerot									
0	1	2	3	4	5	6	7	8	9
4	7	6	6	7	7	6	6	6	6
Pienet kirjaimet yhteensä									172
Kaikki yhteensä									437

Taulukko 5: Prototyyppien merkkikohtainen jakauma jälkimmäisissä kokeissa.

järjestelyssä.

4.2.3 Kokeiden tulokset

Parhaat tulokset ja tarkemmat tiedot luokittelijoista ovat taulukossa 6. SS on symbolijonoihin perustuva luokitin, jossa etäisyysero otetaan huomioon vain diskretointirajan mukaisesti, eli kaikki symbolit vastaavat samanpituisia vektoreita merkin koordinaattiavaruudessa. LSS taas on symbolijonoluokitin, jossa suoraa etäisyysinformaatiota on käytetty.

Tuloksista nähdään, että kumpikaan luokittimista ei suoriudu tästä tehtävästä parametrien optimoinnin jälkeenkään lähestulkoonkaan niin hyvin kuin olisi toivottavaa. Lähes 30 prosentin virhe ei ole hyväksyttävää pienillä merkeillä, ja vain hieman yli 50 prosentin tunnistustulos kaikilla merkeillä luokittimen paremmalla versiolla on mitään käyttöä ajatellen kohtuuttoman huono.

Luokittelija	**SS**	**SS**	**LSS**	**LSS**
Merkkijoukko	a-z	a-öA-Ö0-9	a-z	a-öA-Ö0-9
Prototyyppien kokonaismäärä	172	437	172	437
Kynän noston huomionti	on	on	on	on
Kulmien havainnointi	on	on	on	on
Pituusinformaatio	ei	ei	on	on
Suuntien lukumäärä d	16	8	16	32
Diskretointivälin pituus ℓ	15	3	15	15
Korvauksen lisähinta κ_1	ei	ei	0.6	0.45
Lisäyksen lisähinta λ_1	ei	ei	0.2	0.65
Poiston ja lisäyksen hintojen suhde ρ	1	1	1	0.5
Kynä-ylhäällä korvauksen hinta κ_2	ei	ei	5	1
Kynä-ylhäällä lisäyksen hinta λ_2	ei	ei	5	1
Paras tulos	**28.2**	**50.1**	**27.6**	**47.4**

Taulukko 6: Jälkimmäisessä kokeessa käytettyjen omien luokittimien parhaat tulokset.

Merkit	Kirjoittajia	Lukumäärä
a-öA-Ö0-9	8	3081

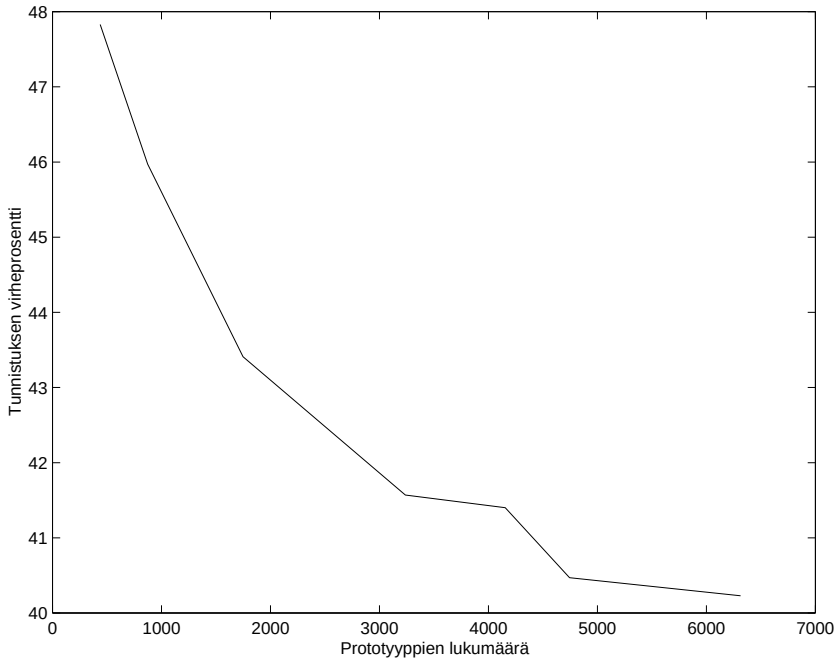
Taulukko 7: Prototyyppien määrän vaikutuksen tarkastelussa käytetyn datajoukon tiedot.

4.2.4 Prototyyppien määrän vaikutus

Selvästi ensimmäisiä kokeita huonommat virheprosentit johtuvat suurelta osin prototyyppien määrän vähyydestä jälkimmäisissä kokeissa ensimmäisiin verrattuna. Tämä havaitaan kuvassa 3 esitettävästä tunnistusprosentin käyttäytymisestä prototyyppien lukumäärän kasvaessa.

Nämä kokeet suoritettiin kaikille merkeille ja samoilla parametrien arvoilla kuin taulukossa 6 olevalla kaikki merkit käsittelevä LSS. Prototyyppien muodostamiseen on käytetty samaa data kuin muutenkin toisissa kokeissa, mutta luokiteltavana datana on ollut myöhemmin samalla tarkkuudella TKK:lla kerätty merkkijoukko. Tämän merkkijoukon tarkemmat tiedot ovat taulukossa 7. Samoja merkkejä ei esiinny opetus- ja testijoukoissa lainkaan, mistä johtuu virheprosentin hieman korkeampi lähtötaso. Kuvasta 3 havaitaan siis, että prototyyppien määrän kasvattaminen laskee virheprosenttia selvästi.

Kuitenkin näissä kokeissa lopullinen virhetaso prototyyppejä ollessa 6313 kappaletta oli 40.2%, kun ensimmäisessä koesarjassa päästiin 35.3%:n virheeseen 7231:llä prototyypillä.



Kuva 3: Virheprosentin käyttäytyminen prototyyppien lukumäärän kasvaessa.

5 Vertailu muihin järjestelmiin

Vertailutuloksena on käytetty point-to-point [5], eli pisteittäisellä sovituksella laskettuja tuloksia, koska nämä tulokset ovat kyseisellä järjestelmällä parhaita. Tuloksia tarkasteltaessa on kuitenkin syytä pitää mielessä ero näiden kahden laskentamenetelmän tehovaatimuksissa, laskutoimitusten määrä on huomattavasti suurempi suoritettaessa *dynamic time warping*:ia pisteittäin kuin laskettaessa huomattavasti vähempää määrää differenssejä.

Tulokset on esitetty ensimmäiselle datalle taulukossa 8 ja jälkimmäiselle datalle taulukossa 9. PP viittaa laboratoriossa tehtyyn, yllämainittuun point-to-point-metriikkaan perustuvaan luokitteluun. Ensimmäisissä kokeissa kaikilla luokittelijoilla on käytetty samaa prototyyppijoukkoa, toisessa on tehty klusterointi PP-metriikan mukaisesti samasta lähtöjoukosta kuin SS ja LSS:llä. LSS on yllä esitelty symbolijonomenetelmä ja REF-SS on Jari Kankaan matlabilla tekemä symbolijonoluokitin, joka suorittaa *dynamic time warping*:in symbolijonoille.

Symbolijonoja voidaan muodostaa erittäin monella eri tavalla ja myös esikäsittelyn vaikutus tuloksiin on kiistämätön. Nämä kokeet ovat kuitenkin osoittaneet, että symbolijonoilla saadut tulokset häviävät selvästi point-to-point

	PP	REF-SS	LSS
a-z	10.4	16.2	13.2
a-öA-Ö0-9	19.4	30.5	35.3

Taulukko 8: Vertailutuloksia ensimmäisen kokeen datalla.

	PP	LSS
a-z	6.8	27.6
a-öA-Ö0-9	18.2	47.4

Taulukko 9: Vertailutuloksia jälkimmäisen kokeen datalla.

etäisyysmitan antamille tuloksille. On kuitenkin huomioitava myös selvä ero laskentakapasiteetin tarpeessa: symbolijonoilla tarvittava laskenta-aika vähenee huomattavasti.

6 Johtopäätökset

Jos oletetaan käytössä olevan laskentakapasiteetin olevan rajoitettu, näyttäisi symbolijonojen käyttö olevan harkinnan arvoinen ratkaisu. Tuloksista voidaan huomata, että käyttämällä suoraa pisteidenvälistä etäisyyttä päästään loppujen lopuksi selvästi parempiin tuloksiin, mutta suhteutettaessa tämä tulosten ero käytettävissä olevan datan määrään sekä tehtävään laskentatyömäärään, voi symbolijonojen käyttö joissain tilanteissa olla varteenotettava vaihtoehto. Mutta laskentakapasiteetin ollessa riittävä, täytyy todeta esimerkiksi point-to-point [5] etäisyyden antavan todella paljon parempia tuloksia.

Selvä ero tuloksissa kahdella eri datajoukolla tehtyjen kokeiden välillä johtunee suurelta osin käytettyjen prototyyppien määrästä. Ensimmäisessä kokeessa prototyyppijoukko oli selvästi suurempi kuin testijoukko, kun taas jälkimmäisessä kokeessa prototyyppijoukon koko oli vain murto-osa testijoukon koosta. Tehdyt kokeet osoittavat, että symbolijonojen käyttö vaatii toimiakseen suuremman määrän prototyyppejä. Symbolijonoihin perustuvan tunnistuksen laskennallinen keveys toisaalta tekee myös mahdolliseksi suuremman prototyyppijoukon käytön.

Toinen mahdollinen syy ensimmäisellä datajoukolla saatujen tulosten paremmuuteen on kirjoittajien lukumäärä opetusjoukossa. Ensimmäisessä koesarjassa kirjoittajia oli opetusjoukkossa 114, kun taas toisessa koesarjassa heitä oli vain 22.

Viitteet

[1] T.M. Cover and P.E. Hart. Nearest neighor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27, 1967.

[2] J. Laaksonen, J. Hurri, E. Oja, and J. Kangas. Comparison of adaptive strategies for on-line character recognition. In *Proceedings of International Conference on Artificial Neural Networks*, pages 245–250, 1998.

[3] J. Laaksonen, J. Hurri, E. Oja, and J. Kangas. Experiments with a self-supervised adaptive classification strategy in on-line recognition of isolated handwritten latin characters. In *Proceedings of sixth International Workshop on Frontiers in Handwriting Recognition*, pages 475–484, August 1998.

[4] D. Sankoff and J. B. Kruskal. *Time Warps, String Edits, and Macromolecules: The Theory and Practise of Sequence Comparison.* Addison-Wesley, 1983.

[5] V. Vuori. Adaptation in on-line recognition of handwriting. Master's thesis, Helsinki University of Technology, 1999.

[6] H. Yuen. A chain coding approach for real-time recognition of on-line handwritten characters. In *Proceedings of International Conference on Acoustics, Speech, and Signal Processing*, volume 6, pages 3426–3429, 1996.